I claim:

15

- 1. A method for compressing a Rabin signature, s, for a user having a public key, n, comprising the step of:
- generating a compressed Rabin signature based on a continued fraction expansion of s/n.
 - 2. The method of claim 1, wherein said continued fraction expansion of s/n further comprises the steps of
- 10 computing principal convergents, u_i/v_i, for i equal to 1 to k, of a continued fraction expansion of s/n, where k is a largest integer for which principal convergents are defined;

establishing an index l, such that $v_l < \sqrt{n} \le v_{l+1}$; and generating a compressed Rabin signature (v_l, m) for a message, m.

3. A method for compressing a Rabin signature, s, for a message, m, and a user having a public key, n, comprising the steps of:

computing principal convergents, u_i/v_i , of a continued fraction expansion of s/n;

- 20 establishing an index l, such that $v_l < \sqrt{n} \le v_{l+1}$; and generating a compressed Rabin signature (v_l, m) .
 - 4. The method according to claim 3, wherein $sv=u \pmod{n}$.
- 25 5. The method according to claim 3, wherein $|v| \le \sqrt{n}$.
 - 6. The method according to claim 3, wherein $|u| \le \sqrt{n}$.
- The method according to claim 1, wherein said principal convergents, u_i/v_i,
 are computer for i equal to 1 to k, where k is a largest integer for which principal convergents are defined.

8.	A method for decompressing a compressed Rabin signature (v, m) for a
message, m, a	nd user having a public key, n, comprising the steps of:
	applying a message formatting function, h, to the message, m, to computing
h(m);	
	computing a value, t, as h(m)v ² mod n;
	obtaining a value, w, as a square root of the value, t;
	computing a signature value, s, as w/v mod n; and

10

5

- 9. The method of claim 8, further comprising the step of generating an error if no integer square root exists.
- 10. A method for compressing an RSA signature, s, for a message, m, and a user having a public key (n, e), comprising the steps of:

providing a decompressed signature (s,m).

computing principal convergents, u_i/v_i , of the continued fraction expansion of s/n;

establishing an index l, such that $v_l < n^{(1-l/e)} \le v_{l+1}$; and generating a compressed signature (v_l, m) .

20

11. A method for decompressing a RSA signature (v, m) for a message, m, and a user having a public key (n, e), comprising the steps of:

applying a message formatting function, h, to the message, m, to computing h(m);

computing a value, t, as h(m)v^e mod n;
determining whether the values t or t-n have an eth root over integer values;
computing a value, w, as the eth root; and
computing the decompressed signature (w/v mod n, m).

12. The method of claim 11, further comprising the step of generating an error 30 if no eth root exists.

13. A system for compressing a Rabin signature, s, for a user having a public key, n, comprising:

a memory; and

at least one processor, coupled to the memory, operative to:

- 5 generate a compressed Rabin signature based on a continued fraction expansion of s/n.
 - 14. The system of claim 13, wherein said processor is further configured to perform said continued fraction expansion of s/n by:
- 10 computing principal convergents, u_i/v_i, for i equal to 1 to k, of a continued fraction expansion of s/n, where k is a largest integer for which principal convergents are defined;

establishing an index l, such that $v_l < \sqrt{n} \le v_{l+1}$; and generating a compressed Rabin signature (v_l, m) for a message, m.

15

20

25

15. A system for decompressing a compressed Rabin signature (v, m) for a message, m, and user having a public key, n, comprising:

a memory; and

at least one processor, coupled to the memory, operative to:

apply a message formatting function, h, to the message, m, to computing h(m);

compute a value, t, as h(m)v² mod n; obtain a value, w, as a square root of the value, t; compute a signature value, s, as w/v mod n; and providing a decompressed signature (s,m).

16. The system of claim 15, wherein said processor is further configured to generate an error if no integer square root exists.